

Japanese Patent Laid-open No. 2000-148513 A

Publication date : May 30, 2000

Applicant : Matsushita Denki K.K.

Title : Task control method and task control device

[0005] The present invention has been achieved in view of the problems inherent in the conventional art, and an object thereof is to provide a task control method and a task control device capable of naturally assigning a low priority task to a CPU to improve through-put of a real time OS even when a high priority task continues to dominate the CPU.

[0047] Firstly, the timer-value control unit 103 receives a 1-time slice signal from the system clock 111 and counts down the timer value which is task information of each task on the executable queue 141 (step 301). As long as the timer value indicates a specific value such as "-1" as described above, the count is not decremented. Subsequently, it is determined whether a task of which timer value is "0" exists. When there is no such task, the process returns to a state waiting for the reception of the 1-time slice signal (step 301) (step 311). When the task of which timer value is "0" exists, the priority of the task is changed to the time out priority (step 321), the executed-task switch operation shown in Fig. 4 is performed, and the process returns to a state waiting for the reception of the 1-time slice signal (step 301).

(19)日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開2000-148513

(P2000-148513A)

(43)公開日 平成12年5月30日 (2000.5.30)

(51)Int.Cl.<sup>7</sup>

G 0 6 F 9/46

識別記号

3 4 0

F I

G 0 6 F 9/46

マーク〇(参考)

3 4 0 B 5 B 0 9 8

審査請求 有 請求項の数8 OL (全10頁)

(21)出願番号 特願平10-320759

(22)出願日 平成10年11月11日 (1998.11.11)

(71)出願人 000005821

松下電器産業株式会社

大阪府門真市大字門真1006番地

(72)発明者 森久 謙二郎

神奈川県横浜市港北区綱島東四丁目3番1

号 松下通信工業株式会社内

(74)代理人 100105050

弁理士 鷲田 公一

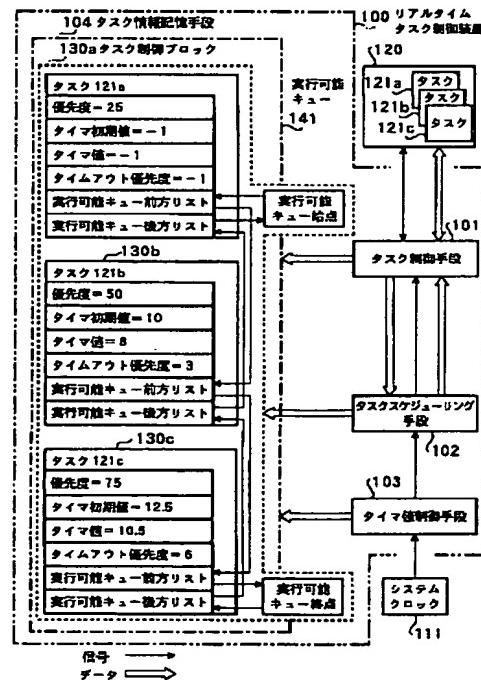
Fターム(参考) 5B098 C003

(54)【発明の名称】 タスク制御方法およびタスク制御装置

(57)【要約】

【課題】 高優先度タスクがCPUを独占し続いている場合でも、低優先度タスクを効率的にCPUに割り当て、システム全体のスループットを向上させること。

【解決手段】 タスク制御プロック 130a～130cには、パラメータとして、初期優先度、タイム初期値、タイムアウト優先度が登録される。タイムアウト優先度は初期優先度よりも高優先度である。タイム値制御手段 103は、タイム値をカウントダウンしていく、タイムアップすると、タスクの優先度を初期優先度からタイムアウト優先度に変更する。タスクスケジューリング手段 102は、変更された優先度にしたがって実行可能キュー141のソーティングを行い、タスク制御手段 101は、そのソーティング順に実行るべきタスクの切換を行う。



## 【特許請求の範囲】

【請求項1】 実時間処理を必要とする複数のタスクの各々に優先度を付与し、その優先度に基づき優先判定を行ってタスクを実行する場合に、前記複数のタスクのうちの少なくとも一部のタスクについて、所定期間内に実行されなかつた場合にはそのタスクの優先度を高める優先度変更処理を施すことを特徴とするタスク制御方法。

【請求項2】 あらかじめタスクに、初期優先度と、この初期優先度が適用される期間と、前記初期優先度よりも高位のタイムアウト優先度とを設定しておき、前記初期優先度が適用される期間が経過すると、前記タイムアウト優先度により優先判定を行う請求項1記載のタスク制御方法。

【請求項3】 実時間処理を必要とする複数のタスクの各々に優先度を付与し、その優先度に基づき優先判定を行ってタスクを実行するタスク制御方法において、タスクが実行可能状態になると、そのタスクについて初期優先度、タイム初期値およびタイムアウト優先度を取得して実行可能キューに登録し、時間の経過と共にタイム値を前記タイム初期値よりカウントダウンしていく、前記タイム値が「0」になるとそのタスクの優先度を前記初期優先度から前記タイムアウト優先度に変更して前記実行可能キューのソーティングを行い、前記実行可能キューのソーティングの結果として実行可能キューの先頭タスクが前回のソーティング時における先頭タスクと異なるものとなった場合、前記実行可能キューの前記先頭タスクを実行タスクとしてCPUに割り当てる特徴とするタスク制御方法。

【請求項4】 タスク生成時に、前記初期優先度、タイム初期値、タイムアウト優先度の値を設定することを特徴とする請求項3記載のタスク制御方法。

【請求項5】 前記タイムアウト優先度の範囲が前記初期優先度の取り得る値の範囲外であり、かつ、前記タイムアウト優先度の方が前記初期優先度よりも高位であることを特徴とする請求項3記載のタスク制御方法。

【請求項6】 前記実行可能キューに登録されているタスクのタイム値が所定の値の場合、前記タイム値のカウントダウンが行われず、これによってそのタスクの優先度は、常に前記初期優先度となることを特徴とする請求項3記載のタスク制御方法。

【請求項7】 タスクの状態変化の検出および実行タスクの切換を行うタスク制御手段と、このタスク制御手段によって状態変化が検出されたタスクに関する、初期優先度、タイム初期値およびタイムアウト優先度を含む情報を記憶するタスク情報記憶手段と、実行可能キューに登録されているタスクを高優先度順にソーティングするタスクスケジューリング手段と、前記実行可能キューに登録されているタスクに関するタイム値を前記タイム初期値からカウントダウンしていく、前記タイム値が

「0」となった場合に、そのタスクの優先度を前記初期

優先度からタイムアウト優先度に変更すると共に、前記タスクスケジューリング手段に実行可能キューのソーティングを要求するタイマ値制御手段と、を具備することを特徴とするタスク制御装置。

【請求項8】 前記タスクスケジューリング手段は、実行可能キューのソーティング時に優先度の同じタスクが複数存在する場合、タスクの優先度を初期優先度からタイムアウト優先度に変更したタスクをキューの高位、あるいはキューの低位にソーティングすることを特徴とする請求項7記載のタスク制御装置。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】本発明はタスク制御方法およびタスク制御装置に関し、特に、複数の実時間処理を必要とするタスクの実行を制御するリアルタイムオペレーティングシステム（以下リアルタイムOSと記述する）を用いたタスク制御方法およびタスク制御装置に関する。

## 【0002】

20 【従来の技術】タスクスケジューリング方法の典型的なものは、タスクに優先度を設定しておき、タスクがCPUに割り当てられるのを待っている状態（以下、実行可能状態という）のときに、優先度の高位のものから順に実行可能キューに登録（ソーティング）し、実行可能キューの先頭タスクから順に実行するという方法である。

## 【0003】

【発明が解決しようとする課題】しかし、従来のスケジューリング方法においては、高優先度の特定のタスクが頻発した場合や、複数のタスク実行が重なった場合などには、低優先度のタスクは、いつ実行されるか全く見当がつかず、最悪の場合には、全く実行されない、という問題を有していた。

30 【0004】この問題を解決する方法としては、割り込みを利用して、低優先度のタスクを強制的にCPUに割り当てるという手法が考えられる。つまり、タスクが実行可能状態に変化したら割り込みタイムをスタートさせ、設定した時間が経過するとアプリケーションレベルで割り込みをかけてタスクを実行する、というものである。しかし、この方法では高優先度の特定のタスクが頻発していない場合や、複数のタスクの実行が重なっていない場合でも、一律に割り込みタイムの設定時間の経過後でなければタスク実行がされず、結果としてシステム全体のスループットを低下させる恐れがある。

40 【0005】本発明は、このような従来技術の問題点を解決するためになされたものであり、高優先度タスクがCPUを占有し続けている場合でも、低優先度タスクをCPUに無理なく割り当てるリアルタイムOSのスループットを向上させることができるタスク制御方法およびタスク制御装置を提供することを目的とする。

## 【0006】

【課題を解決するための手段】本発明のタスク制御方法では、あらかじめアプリケーションが設定した時間をしきい値とし、そのしきい値の時間が過ぎると、タスクの優先度を高位の優先度に変化させ、そのタスクがCPUに割り当てられる確率を高める。

【0007】これにより、高優先度のタスクがCPUを占有し続けている場合であっても、低優先度のタスクが実行される可能性が格段に高まる。しかも、アプリケーションが設定した時間が過ぎると優先度がダイナミックに変化することから、タスクが実行可能状態に変化してから実際にCPUに割り当てられるまでの時間をアプリケーションが設定した値程度に保証することが可能となる。また、特別なパラメータや割り込み等の他の特別な処理を用いるわけではないので、処理が複雑化しない。すなわち、リアルタイムOSがサポートしている優先度順に処理するという基本的な機能をそのまま利用して低優先度のタスクの処理確率を高める、という手法をとるので、無理がなく、しかも、設定するパラメータは、時間しきい値と、しきい値の前後における優先度だけよいため、処理手順や装置構成が複雑化しない。したがって、システムのスループットの向上を、無理なく、簡素化された方法で実現することができる。

【0008】また、本発明のタスク制御装置は、実行可能キューに登録されているタスクを優先度順にソーティングするタスクスケジューリング手段と、実行可能キューに登録されているタスクについてタイムアウトを検出すると優先度を高位に変更すると共に、タスクスケジューリング手段に実行可能キューのソーティングを要求するタイマ値制御手段と、を有する。

【0009】タイマ値制御手段とタスクスケジューリング手段の働きにより、低優先度のタスクについても、時間経過に伴なうダイナミックな優先度の変更が可能となり、リアルタイムOSがサポートするシステム全体としてのスループットが、無理なく向上する。

#### 【0010】

【発明の実施の形態】本発明のタスク制御方法の第1の態様では、実時間処理を必要とする複数のタスクの各々に優先度を付与し、その優先度に基づき優先判定を行ってタスクを実行する場合に、前記複数のタスクのうちの少なくとも一部のタスクについて、所定期間に内に実行されなかつた場合にはそのタスクの優先度を高める優先度変更処理を施す。

【0011】これにより、低優先度のタスクが予め設定した時間程度で実行される可能性が高まり、無理なくシステム全体のスループットを高めることができる。

【0012】また、本発明のタスク制御方法の第2の態様では、第1の態様において、あらかじめタスクに、初期優先度と、この初期優先度が適用される期間と、前記初期優先度よりも高位のタイムアウト優先度とを設定しておき、前記初期優先度が適用される期間が経過する

と、前記タイムアウト優先度により優先判定を行うようにした。

【0013】したがって、簡単なパラメータの設定によってタスクの優先度変更処理を実現できる。

【0014】本発明のタスク制御方法の第3の態様では、実時間処理を必要とする複数のタスクの各々に優先度を付与し、その優先度に基づき優先判定を行ってタスクを実行するタスク制御方法において、タスクが実行可能状態になると、そのタスクについて初期優先度、タイマ初期値およびタイムアウト優先度を取得して実行可能キューに登録し、時間の経過と共にタイマ値を前記タイマ初期値よりカウントダウンしていく、前記タイマ値が「0」になるとそのタスクの優先度を前記初期優先度から前記タイムアウト優先度に変更して前記実行可能キューのソーティングを行い、前記実行可能キューのソーティングの結果として実行可能キューの先頭タスクが前回のソーティング時における先頭タスクと異なるものとなった場合、前記実行可能キューの前記先頭タスクを実行タスクとしてCPUに割り当てるようにした。

【0015】これにより、高優先度タスクがCPUを占有し続けている場合でも、低優先度タスクをCPUに割り当てることができ、また、低優先度タスクが実行可能状態に変化してから実際にCPUに割り当てられるまでの時間をアプリケーションが設定した値程度に保証することができる。

【0016】本発明のタスク制御方法の第4の態様は、タスク生成時に、前記初期優先度、タイマ初期値、タイムアウト優先度の値を設定する。

【0017】これにより、アプリケーションレベルで簡単にタスク制御を行える。

【0018】本発明のタスク制御方法の第5の態様では、前記タイムアウト優先度の範囲が前記初期優先度の取り得る値の範囲外であり、かつ、前記タイムアウト優先度を前記初期優先度よりも高位とした。

【0019】これにより、タイムアウトして待ち時間が長いタスクが、タイムアウトしていない他のタスクよりも優先的に実行される。

【0020】また、本発明のタスク制御方法の第6の態様では、前記実行可能キューに登録されているタスクのタイマ値が所定の値の場合、前記タイマ値のカウントダウンが行われず、これによってそのタスクの優先度は、常に前記初期優先度となるようにした。

【0021】これにより、タスク生成時に、アプリケーションが各タスクに対して個別にタイムアウト時の優先度変更機能を使用するかしないかを適宜、選択することができる。

【0022】また、本発明のタスク制御装置にかかる第7の態様では、タスクの状態変化の検出および実行タスクの切換を行うタスク制御手段と、このタスク制御手段によって状態変化が検出されたタスクに関する、初期優

先度、タイマ初期値およびタイムアウト優先度を含む情報を記憶するタスク情報記憶手段と、実行可能キューに登録されているタスクを高優先度順にソーティングするタスクスケジューリング手段と、前記実行可能キューに登録されているタスクに関するタイマ値を前記タイマ初期値からカウントダウンしていく、前記タイマ値が「0」となった場合に、そのタスクの優先度を前記初期優先度からタイムアウト優先度に変更すると共に、前記タスクスケジューリング手段に実行可能キューのソーティングを要求するタイマ値制御手段と、を設ける構成とした。

【0023】これにより、タイマ値の制御による優先度の変更を行う手段と、優先度にしたがつた実行可能キューのソーティングを行う手段とをもつ簡素化された構成でもって、高優先度タスクのみならず、低優先度タスクにも、ほぼ所望の範囲でCPUへの割り当てのチャンスが与えられるシステムを構築できる。

【0024】また、本発明のタスク制御装置にかかる第8の態様では、第7の態様において、前記タスクスケジューリング手段は、実行可能キューのソーティング時に優先度の同じタスクが複数存在する場合、タスクの優先度を初期優先度からタイムアウト優先度に変更したタスクをキューの高位、あるいはキューの低位にソーティングする構成とした。

【0025】これにより、優先度が同じタスクが競合した場合でも、不都合なくタスクの処理を続行することができる。

【0026】以下、本発明の実施の形態について図面を参照して具体的に説明する。

【0027】(発明の実施の形態) 図1はリアルタイムタスクの制御を行うタスク制御装置のブロック図である。

【0028】本実施の形態にかかるリアルタイムタスク制御装置100は、例えばタスク121a、タスク121bおよびタスク121c等から構成されるタスク群120を制御するものである。

【0029】このリアルタイムタスク制御装置100は、タスク制御手段101と、タスクスケジューリング手段102と、タイマ値制御手段103と、タスク情報記憶手段104とを有する。

【0030】タスク制御手段101は、制御対象であるタスク群120上の各タスクの状態変化の検出およびタスクスイッチ(実行するタスクの切換)等のタスク制御を実行する。

【0031】タスク情報記憶手段は、タスク制御手段101により検出されたタスクの状態に関する情報(少なくともタスク状態、優先度、タイマ初期値およびタイムアウト優先度を含む情報)を、各タスクに対応して設けられたタスク制御ブロックに記憶する。すなわち、タスク121a、121b、121cに関するタスク情報は

それぞれ、タスク制御ブロック130a、130b、130cに記憶される。ここで、タスク制御ブロック130a～130cに必要な情報が記憶されるということは、実行可能状態となっている全部のタスクが把握され、しかも、どのタスクから順に実行すべきかが、各タスクの優先度を比較することによって一義的に定まるこことを意味する。本明細書では、優先度にしたがつて順序づけられた、実行可能なタスクに関する情報を実行可能キュー141として把握する。つまり、タスク情報記憶手段104は、実行可能キュー141を記憶する手段として位置づけられることになる。

【0032】タスクスケジューリング手段102は、タスク制御手段101によって状態変化が検出されたタスクについて、優先度、タイマ値およびタイムアウト優先度を含むタスク情報を取得し、実行可能キュー141(つまり、タスク制御ブロック130a～130c)に登録し、かつ、高優先度順にソーティング(並び替え)する働きをする。

【0033】図1の装置において、高優先度の順にソーティング(並び替え)することは、タスク制御ブロック130a～130cの各々の実行可能キュー前方リストに、そのタスクよりも一つ下位の優先度をもつ次のタスクの情報の格納アドレスを書き込むこと、あるいは、実行可能キュー後方リストに、そのタスクより一つ上位の優先度をもつ次のタスクの情報の格納アドレスを書き込むこと、を意味する。つまり、タスク130a～130cの現在の優先度はそれぞれ、「25」、「50」、「75」であり、タスク130aが最優先のタスクであり、次に優先されるのがタスク130bであり、次がタスク130cである。したがつて、実行可能キューのソーティングがなされると、図1に矢印で示されるような、優先度にしたがつた順序付けがさなれることになる。

【0034】タイマ値制御手段103は、システムクロック生成手段111から供給されるシステムクロックを用いてタイムスライス毎に(つまり、システムクロックの入力タイミング毎に)、実行可能キュー141内の各タスク制御ブロックのタイマ値を、タイマ初期値からカウントダウンしていく。そして、タイマ値が「0」となった場合、そのタスクの優先度をタイムアウト優先度に変更すると共に、タスクスケジューリング手段102に、実行可能キュー141のソーティング(並び替え)を要求する。

【0035】なお、図1において、タスク121aに対応するタスク制御ブロック130aのタイマ初期値(およびタイマ値)が「-1」となっているが、この「-1」が設定されると、タイマ値のカウントダウンによる優先度変更処理が行われない。つまり、アプリケーションがタスクの優先度を固定したいと判断した場合には、タイマ初期値に「-1」をセットすることによって、簡

単に、優先度変更処理を禁止することができる。

【0036】以上のように構成されたリアルタイムタスク制御装置100の動作とその効果について、図2～図8を参照して具体的に説明する。なお、図9～図12は、比較例(従来例)の動作と効果を示す図である。

【0037】リアルタイムタスクの制御動作は、大きく分けて図2に示すタスクスケジューリング動作と、図3に示すタイマ値制御動作と、図4に示す実行タスクスイッチ動作の3つから構成される。なお、実行タスクスイッチ動作は、タスクスケジューリング動作およびタイマ値制御動作の一連の手順に包含されるものである。また、タイマ値制御動作は、本発明における核となる部分である。

【0038】図1に示すように、実行可能状態のタスクは、タスク121a～121cの3つである。そして、アプリケーションは、タスク生成時にパラメータとして少なくとも優先度、タイマ初期値、タイムアウト優先度の3つの値を設定する。優先度の設定に関し、例えば、アプリケーションが設定するタイムアウト優先度の範囲を0から9とし、同じくアプリケーションが設定する初期優先度を10から99として、初期優先度とタイムアウト優先度の範囲が重ならないように別々に設けることもできる。なお、本実施の形態では、優先度の数値が低いほど優先順位が高い。

【0039】まず、タスクスケジューリング動作について説明する。

【0040】図2に示すように、まず、タスク制御手段101が、制御対象であるタスク群120上の各タスクに対してタスク状態の変化を検出する(ステップ201)。そして、検出したタスク状態変化に対してタスク生成、タスク削除またはそれ以外の3つの場合分けを行う(ステップ211)。

【0041】タスクが新たに生成された場合は、タスク情報記憶手段104が、生成されたタスクに対するタスク制御ブロック130を確保し(ステップ212)、少なくともタスク状態、優先度、タイマ初期値およびタイムアウト優先度を含むタスク情報をタスク制御ブロック130に書き込んで制御ブロックの編集を行う(ステップ213)。

【0042】一方、タスクが削除された場合は、タスク情報記憶手段104が、削除されたタスクに対するタスク制御ブロック130を解放する(ステップ214)。それ以外のタスク状態変化が起こった場合は、タスク情報記憶手段104が、状態変化したタスクに対するタスク制御ブロック130のタスク状態を変更してブロックの編集を行う(ステップ213)。

【0043】そして、タスク制御手段101は、検出したタスク状態変化を、実行不可能状態から可能状態への変化、実行可能状態から不可能状態への変化、あるいはそれ以外の変化の3つの場合に分ける(ステップ22

1)。ここで、タスクが実行不可能状態であるとは、タスクが休止状態であるか、遅延状態であるか、または事象待ち状態であるかのいずれかの状態であることをいう。

【0044】タスクが実行不可能状態から可能状態に変化した場合は、タスク制御手段101がタスク情報記憶手段104から状態変化したタスクに対するタスク制御ブロック130の優先度、タイマ初期値及びタイムアウト優先度を取得し(ステップ222)、タスクスケジューリング手段102が、この取得したタスク情報と共にタスクを実行可能キュー141に追加し(ステップ223)、図4に示す実行タスクスイッチ動作を行い、タスク制御手段101のタスク状態変化検出(ステップ201)待ち状態に戻る。タスクを実行可能キュー141に追加する際、タイマ初期値はタイマ値に格納する。

【0045】タスクが実行可能状態から不可能状態に変化した場合は、タスクスケジューリング手段102が、実行可能キュー141からそのタスクの情報を削除し(ステップ224)、図4に示す実行タスクスイッチ動作を行い、タスク制御手段101のタスク状態変化検出(ステップ201)待ち状態に戻る。それ以外のタスク状態変化が起こった場合は、タスク制御手段101のタスク状態変化検出(ステップ201)待ち状態に戻る。

【0046】次に、図3を用いて、タイマ値制御動作を説明する。

【0047】まず、タイマ値制御手段103が、システムクロック111から1タイムスライス信号を受信し、実行可能キュー141上の各タスクのタスク情報であるタイマ値をカウントダウンする(ステップ301)。なお、上述のようにタイマ値が「-1」等特定の値となっていると、カウントダウンは行われない。続いてタイマ値が「0」であるタスクが存在するかどうかを判定し、存在しない場合は1タイムスライス信号受信(ステップ301)待ち状態に戻る(ステップ311)。タイマ値が「0」であるタスクが存在する場合は、そのタスクの優先度をタイムアウト優先度に変更し(ステップ321)、図4に示す実行タスクスイッチ動作を行い、1タイムスライス信号受信(ステップ301)待ち状態に戻る。

【0048】次に、図4を用いて、実行タスクスイッチ動作について説明する。

【0049】まず、タスクスケジューリング手段102が、実行可能キュー141上のタスクを優先度の高い順にソーティングする(ステップ401)。すなわち、図1の各タスク制御ブロック130a～130b内の実行可能キュー前方リスト、後方リストのアドレスを書き込んで、各タスクの優先順序を決定する。

【0050】高優先度順にタスクをソーティングする際、優先度の同じタスクが複数存在する場合、タスクが実行可能キューに追加された順にキューの高位にソーテ

イングする方法の他、優先度をタイムアウト優先度に変更した方のタスクをキューの高位にソーティングする方法や、優先度をタイムアウト優先度に変更した方のタスクをキューの低位にソーティングする方法を採用することにより、同じ優先度のタスクが競合する場合にも不都合なくソーティングを行える。

【0051】続いて、タスクスケジューリング記憶手段105は、実行可能キューの先頭タスク（最高優先度タスク）が変化したかどうかを検出し（ステップ41

1）、変化しなかった場合は実行タスクスイッチ動作を終了する。変化した場合は、タスクスケジューリング記憶手段105は、タスク制御手段101に実行タスクスイッチ要求信号を送信し、実行タスクスイッチ要求信号を受信したタスク制御手段101は、タスク情報記憶手段104を用いてタスク制御ブロックの書き換えおよびタスクスイッチの実行時に必要なタスク情報の取得を行い（ステップ421）、タスク群120に対して実行タスクのスイッチを行い（ステップ422）、実行タスクスイッチ動作を終了する。

【0052】このようにして、アプリケーションが設定した所定の時間がタイムアウトすると、低優先度のタスクの優先度がステップ的に変更されて、そのタスクの実行確率が高まる。図1では、タスク121aが最も優先度が高く、続いてタスク121b、121cの順となっていた。ここで、タイムアウトに伴う優先度変更によって、タスク121bの優先度がもっとも高位になり、続いてタスク121c、タスク121aの順になると、実行可能キュー141のソーティング後の状態は、図5のように変化する。このように、所定の時間経過と共にタスクの優先度をダイナミックに変更し、その都度、実行可能キューのソーティングを行うことで、低い優先度のタスクも、アプリケーションが設定した時間程度で実行されることになる。

【0053】本発明の実施の形態によるタスク実行の様子を、図6～図8に示す。

【0054】図6は、3つの周期タスク（タスク121a、タスク121b、タスク121c）の、1周期内におけるタスク実行時間およびタスクの実行周期を示す図である。図7は、図6に示すような3つの周期タスクに対して、図1のタスク制御ブロック130a～130cに示される値の優先度、タイマ初期値、タイムアウト優先度を付与して（つまり、タスク121aに対しては優先度25、タイマ初期値-1、タイムアウト優先度-1を付与し、タスク121bに対しては優先度50、タイマ初期値10、タイムアウト優先度3を付与し、タスク121cに対しては、優先度75、タイマ初期値12.5、タイムアウト優先度6を付与して）、本発明の実施の形態によるタスク実行を行った場合の時間軸に対する各タスクの優先度の変化および実行タスクの変化を示す図である。図8は、図7で示された実行タスクの変化か

ら各タスクのタスク実行待ち時間の最高値を最高タスク実行待ち時間として、合計値をタスク実行待ち時間合計として示すものである。

【0055】図7において、白抜きの三角印はタスクが実行状態に変化したことを示し、黒い三角印は、1周期分の処理を終了して実行不可能状態に変化したことを示す。また、図7の下側の図において、実線の横線はタスクが実行されている期間を示し、点線の横線はタスクが実行待ち状態となっている期間を示している。

【0056】図7の上側の図に示されるように、10タイムスライス（10TS）が経過するとタイムアウトによりタスク121aの優先度がもっとも高くなり、12.5タイムスライス（12.5TS）が経過すると、タイムアウトに伴いタスク121bの優先度がタスク121aに続いて高位となる。これにより、図7の下側に示すように、初期優先度が低くかったタスク121b、121cも、アプリケーションが設定したタイマ初期値（タスク121bについては10TS、タスク121cについては12.5TS）程度の待ち時間で実行される。

【0057】図8の左側に示されるように、タスク121aの待ち時間は6.25TSであり、タスク121bの待ち時間は10TSであり、タスク121cの待ち時間は15TSである。このデータから、各タスクがCPUに効果的に割り当てられ、システム全体としての処理効率が向上しているのがわかる。

【0058】図9および図10は、優先度の変更を行わない従来方式によるタスク実行の様子とその結果を比較例として示すものである。

【0059】図9から明らかなように、低優先度のタスクは上位のタスクの処理が終わるまで、待たされ続けることになる。したがって、図10の左側に示されるように、最も上位のタスク121aは待ち時間が0であるものの、タスク121bについては待ち時間が15、タスク121cについては待ち時間が25となり、図8の本実施の形態の場合と比べ、低優先度のタスクの待ち時間がかなり長い。また、図8と図10の比較から、最高タスク実行待ち時間値およびタスク実行待ち時間の合計値についても、本実施の形態のほうがトータルの値が小さく、効率的な処理がなされていることがわかる。

【0060】図11および図12に、アプリケーションレベルで割り込みタイマを使用した従来例のタスク実行の様子およびその結果を示す。

【0061】割り込みを使用して低優先度のタスクを強制的に実行させる従来例では、CPUが空き状態でも、割り込みが実行されるまで実行タスクの切り替えが行われることはなく、結果としてシステムのスループットを低下させるという不合理な結果が生じることがある。これに対し、本実施の形態によるタスク実行ではCPUが空き状態になればタイマ値が0でなくともCPUへの割

り当てが行われるので、結果としてシステムのスループットを低下させることはない。

【0062】図11においては、時刻21.25TSでCPUが空き状態であるにも関わらずタスク121bは実行されないが、図7では、タスク121bは21.25TSで実行される。

【0063】図8に示されるように、本実施の形態では、タスク121bおよびタスク121cの最高タスク実行待ち時間はタイマ初期値程度に抑えられている。また、タスク121aのタスク実行待ち時間合計も、アプリケーションレベルで割り込みタイマを使用する場合(図12)よりも小さく抑えられていることがわかる。

#### 【0064】

【発明の効果】以上説明したように本発明によれば、リアルタイムタスク制御装置にタイムアウト時の優先度変更機能を設けることにより、高優先度タスクがCPUを占有し続いている場合でも、低優先度タスクをシステム全体のスループットを低下させること無くCPUに割り当て、また、低優先度タスクが実行可能状態に変化してから実際にCPUに割り当てられるまでの時間をアプリケーションが設定した値程度に保証することができる。これにより、システム全体のスループットが向上する。また、時間に関する制御は、システムロックから得られるタイムスライス毎にタイマ値をカウントダウンする処理のみなので、リアルタイムタスク制御を容易に実現することができる。

#### 【図面の簡単な説明】

【図1】本発明の実施の形態におけるリアルタイムタスク制御装置のブロック図

【図2】本発明の実施の形態におけるタスクスケジューリング動作を示すフロー図

【図3】本発明の実施の形態におけるタイマ値制御動作を説明するためのフロー図

10

【図4】本発明の実施の形態における実行タスクスイッチ動作を説明するためのフロー図

【図5】本発明の実施の形態において、タイムアウトに伴なって優先度の変更を行った場合の実行可能キューの状態例を示す図

【図6】本発明の実施の形態における各タスクのタスク実行時間およびタスク実行周期を示す図

【図7】本発明の実施の形態における各タスクの優先度の変化および実行タスクの変化を示す図

【図8】本発明の実施の形態における各タスクの最高タスク実行待ち時間およびタスク実行間待ち時間の合計を示す図

【図9】従来のスケジューリング方法を説明するための各タスクの優先度の変化及び実行タスクの変化を示す図

【図10】従来例における各タスクの最高タスク実行待ち時間およびタスク実行間待ち時間の合計を示す図

【図11】他の従来例における各タスクの優先度の変化および実行タスクの変化を示す図

【図12】他の従来例における各タスクの最高タスク実行待ち時間およびタスク実行間待ち時間の合計を示す図

#### 【符号の説明】

100 リアルタイムタスク制御装置

101 タスク制御手段

102 タスクスケジューリング手段

103 タイマ値制御手段

104 タスク情報記憶手段

105 タスクスケジューリング手段

111 システムロック

120 アプリケーションタスク群

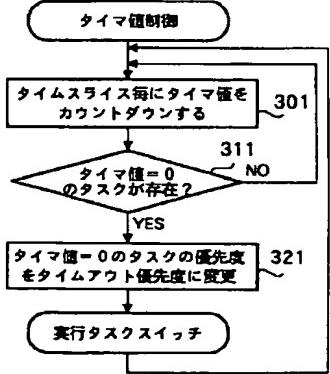
121 アプリケーションタスク

130a～130c タスク制御ブロック

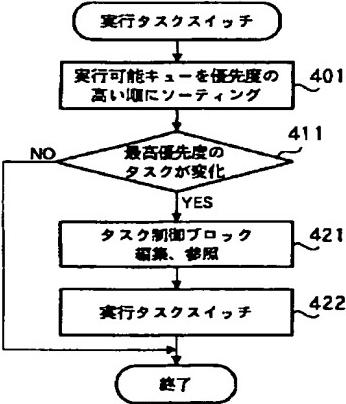
141 実行可能キュー

30

【図3】



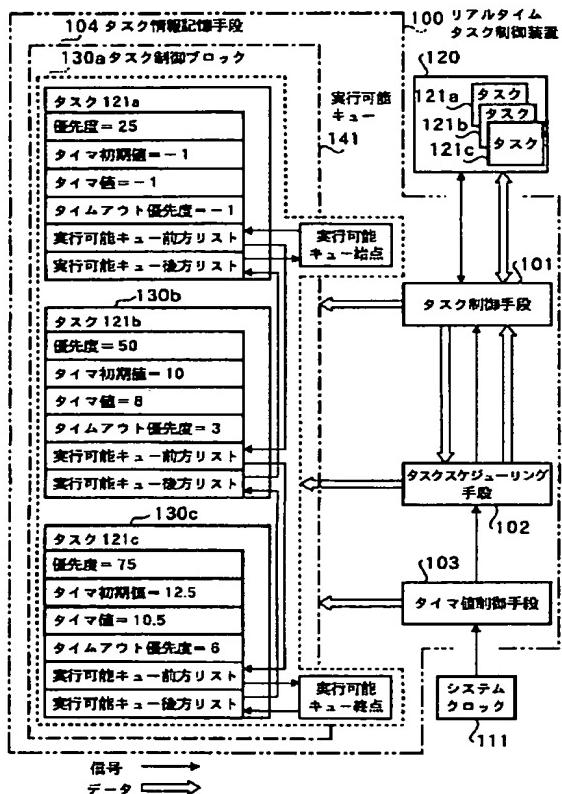
【図4】



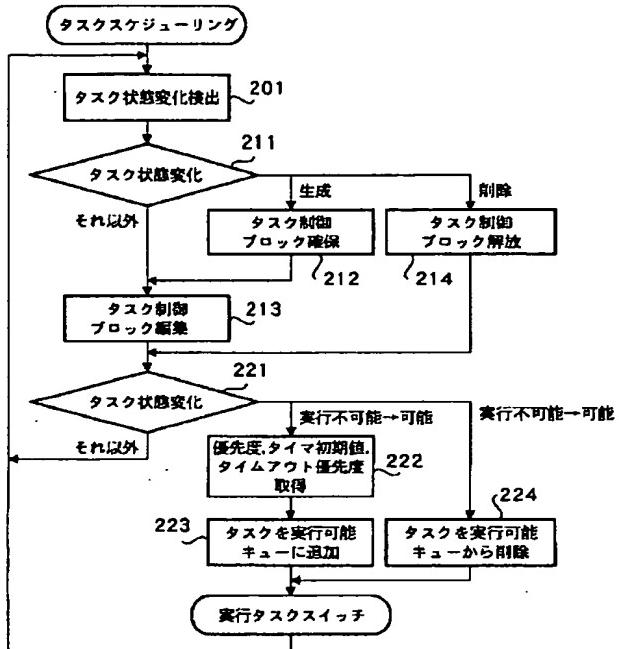
【図6】

タスク	タスク実行時間	タスク実行周期
タスク121a	15	30
タスク121b	5	20
タスク121c	1.25	40

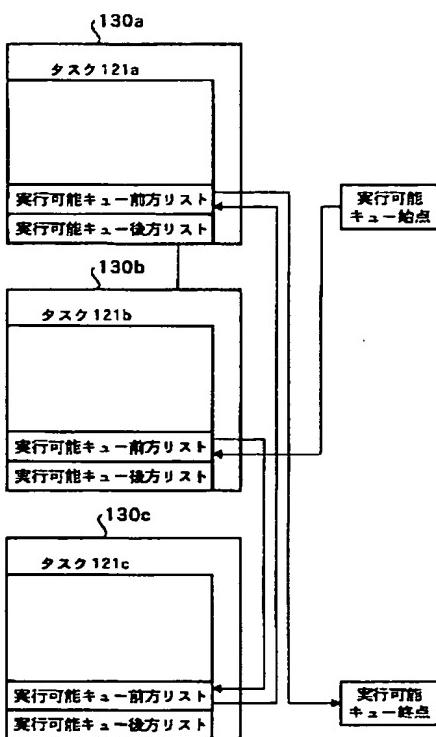
【図1】



【図2】



【図5】



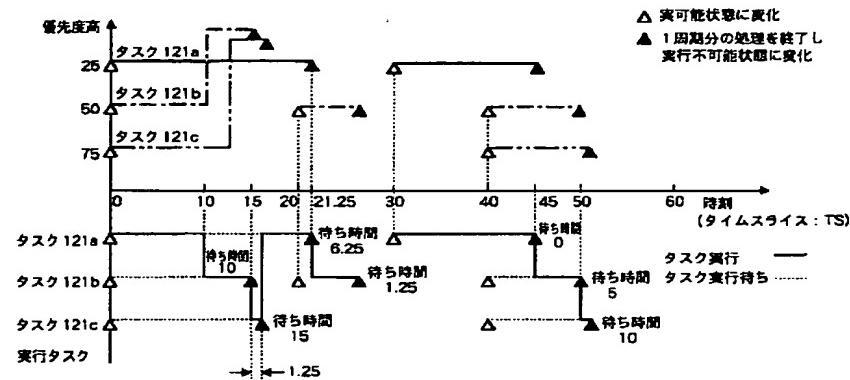
【図8】

タスク	最高タスク実行待ち時間 (TS)	タスク実行待ち時間合計 (TS)
タスク 121a	6.25	6.25
タスク 121b	10	16.25
タスク 121c	15	25
3つのタスクの合計	31.25	47.5

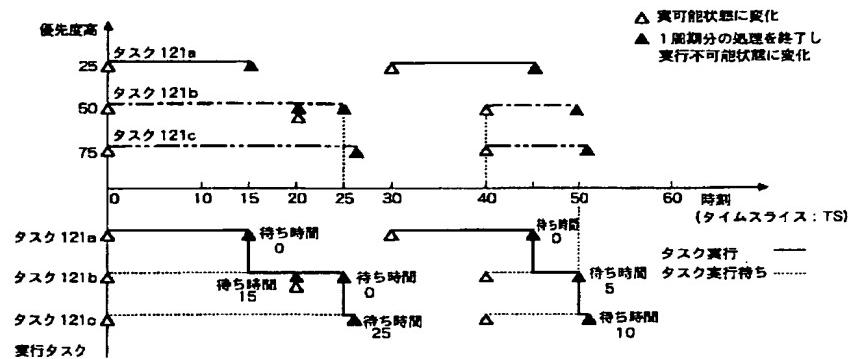
【図10】

タスク	最高タスク実行待ち時間 (TS)	タスク実行待ち時間合計 (TS)
タスク 121a	0	0
タスク 121b	15	20
タスク 121c	25	35
3つのタスクの合計	40	55

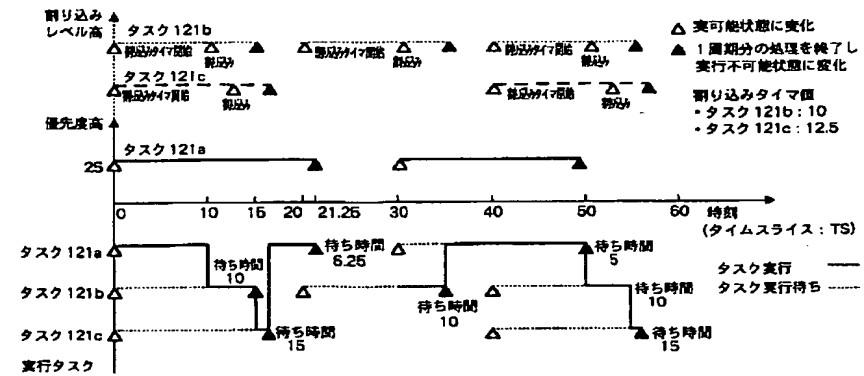
【図7】



【図9】



【図11】



【図12】

タスク	最高タスク実行待ち時間 (TS)	タスク実行待ち時間合計 (TS)
タスク121a	6.25	11.25
タスク121b	10	30
タスク121c	15	30
3つのタスクの合計	31.25	71.25